REMARKS

Claims 1-13, 15-25 and 37 remain pending in the application. Claims 1 and 21-23, the third paragraph on page 13 of the specification, and Fig. 2 have been amended without introduction of new matter. Favorable reconsideration is respectfully requested in view of the above amendments and the following remarks.

Before proceeding with an analysis of the Office Action, the Examiner is thanked for the courtesies extended to the undersigned throughout the personal interview conducted on October 9, 2007.

During the interview, the undersigned and the Examiner discussed the meaning of the term "pipelining" and how this term relates to the applied Green and Wilkinson III et al. documents. During the interview, the Examiner asserted that pipelining includes parallel processing and referred a definition of "pipelining" from the Microsoft Computer Dictionary, Fifth Edition. A copy of that definition is submitted in an information disclosure statement being filed concurrently with this reply.

Also during the interview, the Examiner and the undersigned discussed the rejection under 35 U.S.C. 101. In particular, the undersigned pointed out that each of the claims recite a value associated with the stored key value is output as a result of a look up operation, which is a useful, concrete and tangible result. The Examiner asserted that the result is not claimed in a manner in which it would be persistent, such as in an output memory. While it is not believed an amendment to claims 1 and 21-23 reciting any such limitation would be necessary for compliance under Section 101, the claims have been amended to explicitly recite that an outputted value is stored in a FIFO. Support for this amendment is found in the specification, at least at page 13, lines 26-27 and page 15, lines 28-29, and Fig. 2.

The substance of the remaining items discussed during the interview are addressed in the following:

On pages 2 to 3 of the Action, the drawings were objected to for allegedly failing to show claimed features relating to a plurality of look up state machines connected in parallel with the same look up table. The Action goes on to assert that Fig. 1 is not interrelated with Fig. 2. In response, Applicant traverses this objection because Figure 2 clearly shows state machines 206a-206d connected parallel with the memories 212a, 212b (with an intervening memory arbiter 208) (see, page 11, lines 3-9, and page 12, lines 15-19). It appears the Office mistakenly understood that the look up table of the Figure 1 embodiment is not disclosed as combinable with the Figure 2 embodiment. However, those of ordinary skill in the art would

have understood that the exemplary Figure 1 lookup table is a high-level conceptual diagram showing the operation of both software and hardware, and that this lookup table may be implemented with the exemplary table lookup engine shown in Figure 2. Furthermore, combinations of the features shown in Figures 1 and 2 are described in the original claims and in various parts of the specification (e.g., see page 2, line 40 to page 3, line 9). Because the objection is based on an erroneous assumption, it is believed improper and should be withdrawn.

During the October 9, 2007, personal interview, the undersigned explained why the drawing objection should be rescinded, but suggested a minor amendment to Fig. 2 that would abundantly address the Examiner's concerns. In accordance with this suggestion, a block labeled "look up table" is added to each memory 212a and 212b. Additionally, a bi-directional arrow is added between the memory arbiter 208 and memory 212b to correct an inadvertent error presented in a previously submitted drawing correction. Also, the third paragraph of page 13 of the specification has been changed to include a description corresponding to the elements added to Fig. 2. It is respectfully submitted that these amendments fully address the Examiner's concerns expressed both on pages 2 to 3 and during the interview.

It is noted that the forgoing amendments to the specification and drawings correspond to an exemplary embodiment in which a look up table may be distributed across memories 212a and 212b, and do not explicitly indicate other embodiments in which a lookup table may be provided in other ways, for example, in external memory (e.g., see page 9, lines 19-22).

Claims 1-13, 15-25 and 37 are rejected under 35 U.S.C. § 101, as allegedly being directed to non-statutory subject matter. Specifically, the Office asserts that claims 1 and 21 to 23 do not involve transformation of article or physical object to a different state or thing, they merely recite looking up data. The Office further asserts that independent claims 1 and 21 to 23 do not produce a useful, concrete, and tangible result, but instead merely look up data. This rejection is respectfully traversed.

The Examiner initially asserts that independent claims 1, 21, 22 and 23 do not involve transformation of an article or physical object to a different state or thing because they allegedly merely recite looking up data. However, as explained in *AT&T Corp. v. Excel Communications*, 50 USPQ2d 1452, "The notion of "physical transformation" … is not an invariable requirement, but merely one example of how a mathematical algorithm may bring

about a useful application." Thus, lack of "physical transformation" alone is not determinative of whether claimed subject matter is statutory.

Next, the Office alleges that the independent claims do not produce a useful, concrete, and tangible result because they merely look up data. However, the claimed methods do not involve only disembodied mathematical and intangible concepts. Rather, they are directed to subject matter that produces a useful, concrete, and tangible results in the field of data retrieval. For example, claim 21 is directed to a method that includes, *inter alia*, receiving an input key value, comparing it with a plurality of key values, and outputting the value associated with the stored key value that matches the input key value. The method of claim 21 also involves carrying out these processes of receiving, comparing and outputting concurrently by means of a plurality of look up state machines connected in parallel. As described on pages 1 to 2 in Applicant's specification, for example, routers or switches utilize look up tables to retrieve a route for a received packet. The router accomplishes this by extracting a key in the header field of the packet. The key contains information used to look up the route for the received packet. In the look up operation, the router provides the key as input to a look up table and matches the key with a stored key value associated with a route destination. This is very useful in data transfer and communications, is repeatable, and has real world value. Other exemplary uses include flow tables and control lists.

Similar features related to a process of looking up a value are recited in independent claim 22, but with the caveat that the input key value is divided into a plurality of predetermined portions, which are compared with a plurality of key values. For similar reasons, therefore, claim 22 is also believed to recite statutory subject matter.

Similar concepts are brought out in independent claims 1 and 23, which are respectively directed to a look up engine and a computer system comprising a look up engine. More particularly, each of claims 1 and 23 recite, *inter alia*, that a look up engine comprises a storage means for storing a look up table. The look up table includes a plurality of entries, and each entry comprises a value and an associated key value. In operation, a look up is carried out by outputting a value associated with the stored key value that matches an input key value. As mentioned above, such a look up table is useful in routing tables, flow tables and access control lists, for example. For instance, a router or switch may perform a look up using the look up engine to obtain information relating to the route for a received packet by matching a key value extracted from a data packet with a stored key value, and receiving a return value output associated with the matched stored key value. This look up engine,

therefore, is at least useful in data transfer and communications, is repeatable, and has real world value.

For at least these reasons, it is respectfully submitted that each of independent claims 1 and 21-23, and hence claims depending therefrom, recite combinations of features that produce useful, concrete, and tangible results, and thus recite statutory subject matter. Furthermore, the independent claims have been amended to now recite that the outputted value is stored in a FIFO, which is real tangible memory, and the language relating to "enabling" has been amended to either recite that a look up engine operates "to concurrently perform a look up" or that it is "configured to concurrently perform a look up," which Applicant believes fully address the concerns expressed by the Examiner during the October 9, 2007, personal interview. Accordingly, Applicant respectfully requests withdrawal of the Section 101 rejection.

Claims 1-13, 15-25 and 37 are rejected under 35 U.S.C. § 112, second paragraph, as allegedly being indefinite. In particular, the Office asserted the term "enable" is allegedly indefinite. As pointed out above, however, claims 1 and 21-23 have been amended to address the Office's concerns regarding the term "enable."

The Office Action also alleged that the recited phrase "multiple look ups ... be carried out in the same look up table" renders the claims indefinite because it could cause an overload and the claimed method does not handle such a scenario. However, as described with respect to exemplary embodiments, a *plurality* of memory banks may be accessed. More specifically, one example includes two memory banks in which two memory reads can be done concurrently (e.g., see Fig. 2). Thus, multiple state machines have shared and concurrent access to a lookup table distributed across multiple memory banks. Accordingly, it is respectfully requested that the Office withdraw the rejection of claims 1-13, 15-25 and 37 under the second paragraph of 35 U.S.C. § 112.

On pages 5 to 6, claims 1 and 21-25 are rejected under 35 U.S.C. § 102(b) as allegedly being anticipated by Tock et al. (U.S. Patent No. 6,115,802, henceforth "Tock"). This rejection is respectfully traversed.

With respect to the Tock patent, the Office refers to Fig 3 and appears consider this figure to show "a plurality of look up state machines connected in parallel." However, it is respectfully submitted that Fig. 3 of Tock does not show state machines. Fig. 3 does not even show processors simulating a state machine for doing a lookup. Rather, Tock discloses processors that calculate a hash value for the value being looked up.

It appears the Office is asserting that any system that can change state can be defined as a "state machine." It is respectfully submitted that such an interpretation would be overly broad and unreasonable. Contrary to the Office's interpretation, those of ordinary skill in the field of this invention would have understood that the claimed subject matter related to a state machine is directed to a very specific definition/type of "state machine," namely, a Finite State Machine.

A finite state machine is a well-defined mathematical concept where an object has a limited number of independent states and a fixed number of inputs and outputs. It makes a fixed decision about what state to transition to and what outputs to generate. This is a well-understood method in electronic engineering and computer science as an implementation technique. A person of ordinary skill in the art, therefore, would understand that the term "state machines" as in the claims are finite state machines.

A state machine can be implemented in software and is limited to having a finite number of states, whereas a computer program has a potentially infinite number of states. Therefore, to interpret the term of a state machine so broadly to include any hardware or application that holds any state that can be later changed to another state would be incorrect. As is well known on the field of computer science, Alan Turing came up with something called the Halting Problem that says you cannot prove that a given computer program will finish, basically because it is unbounded, i.e. has a potentially infinite number of states which is the opposite of a state machine. Therefore, it is respectfully submitted that the processors of Tock are not configured as state machines for doing look ups.

The hashing function in Tock could have been written as a state machine, but that would be extremely inefficient because hash functions are simple mathematical functions, for example, of the form: X to the N plus M. Therefore, utilization of a hashing function actually teaches way from use of state machines, as defined by the claims.

For at least the above reasons, it is respectfully submitted that Tock does not teach each and every limitation recited in independent claims 1 and 21-23, and hence also in dependent claims 24 and 25. Accordingly, this rejection under Section 102 should be withdrawn. However, to advance prosecution in this application, Applicant would not object to the Examiner inserting "finite" before "state machines" in each of the independent claims by way of an examiner's amendment if the Examiner otherwise considers the present application allowable.

Claims 1-13, 15-25, and 37 are again rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Greene in view of Wilkinson III et al. (henceforth "Wilkinson"). This rejection is respectfully traversed for reasons already stated in the record, and further for the following reasons:

In the most recent office action, at pages 6 to 9, the Office maintains the previously set forth rejection of claims 1-13, 15-25 and 37 based on the Greene and Wilkinson II et al. patents. The Office basically repeats the reasoning set forth in the final office action dated September 26, 2006, except that it is further asserted that Green processes data concurrently based on the description in column 31, lines 61-64, and column 8, lines 29-35. However, the cited part of column 8 explicitly describes a "pipelined" process in which a second value (i.e., "next DEST_IP value") is applied to the lookup engine following a first applied value ("one DEST_IP value") as soon as the first value has been used to generate an address for the first memory array. Thus, the two values are processed one after the other, and are not processed both concurrently, as claimed. Consequently, Green does not teach this feature as recited in the amended independent claims.

Also, the description relied upon from column 31 of Greene relates to a single key having additional fields (e.g., a packet received by a router as described in the immediately preceding paragraph), and providing additional cascading systems to handle the additional fields. However, it does not describe concurrent processing of multiple search requests, as claimed. In the present response, this feature has been further clarified. For example, claims 1 and 23 each recites inter alia the feature of "the look up engine comprising a plurality of look up state machines connected in parallel and configured to concurrently perform a look up of different respective search requests in the same look up table." Claims 21 and 22 have been amended to recite similar subject matter. It is respectfully submitted that Greene patent does not teach concurrent look ups, as claimed.

At page 10, the Office also asserts "Applicant also teaches pipelining." However, this is not the case in the context of the claimed subject matter related to concurrently performing different respective key value search requests using the same look up table. Pipelining is a very limited form of concurrency - it makes the operations sequentially dependent on one another: a later operation can't finish quickly and "overtake" a later operation. However, in the concurrent state machines of the present invention, operations can start and finish in any order. "Pipelining" occurs in the FIFO, but this is not achieving concurrency. Rather, it is just a means for buffering or queuing up pending requests when all the state machines are

busy. As soon as a state machine is free it will take the next lookup and perform it (in parallel with the other lookups).

During the personal interview on October 9, 2007, the Examiner referred to the definition of "pipelining" provided in the Microsoft Computer Dictionary, Fifth Edition, page 406. However, this definition does not address the specific combination of features related to concurrently performed look ups of different respective search requests in the same look up table, as recited in independent claims 1 and 21-23. It is requested that the Office point to a specific citation of this definition and explain where it teaches or suggests concurrency, as recited in the context of the claims.

As pointed out on page 8 of Applicant's reply dated March 18, 2007, "it is not possible for the Greene search engine to commence a new search while the previous search is incomplete and is still progressing through the pipeline. To be more specific, Applicant submits that it is not possible for the first 16 bits of a new search key to enter the first search stage as the previous search enters the second or third stages because the results of the search in the first stage are employed in the second and subsequent search stages, as discussed above. It is to be remembered that returned data from the first search is passed to the output stage or to the address generator of the second and third stage.

It is therefore respectfully submitted that Greene does not in fact disclose or teach a search engine capable of performing multiple concurrent independent searches in the same table. Although Greene discloses conducting partial searches in portions of a table, these partial searches are themselves pipelined. Moreover, different searches are conducted in pipelined fashion as discussed. In Applicant's respectful opinion, therefore, Greene does not provide a starting point for the present ground of rejection."

At page 7 of the Action, it is again noted that the Office states that "search engines are notoriously well known to comprise multiple state machines to handle multitasking," Applicant's experience is that although multiple state machines are often employed in search engines, it is not the case that multiple state machines are employed *to handle multitasking*. Applicant again requests that the Office supply evidence to support this view, so that Applicant may address the objection if it should still prove necessary to do so.

With respect to the Wilkinson II et al. patent, the Office does not present any new arguments in response to Applicant's arguments presented in the reply dated March 18, 2007. Therefore, Applicant incorporates by reference all arguments regarding Wilkinson II et al. presented on pages 9 to 10 of the March 18, 2007, response.

For all these reasons, neither Greene nor Wilkinson, even considered in combination disclose or teach the claimed subject matter related to multiple, independent, concurrent searches. It is respectfully maintained that there is therefore no incentive or motivation to combine these documents in the manner suggested by the Office. Accordingly, it is respectfully asserted that independent claims 1, 21, 22, and 23, as well as their dependent claims 2-13, 15-20, 24-25 and 37 are patentably distinguishable over the prior art of record, and as such, this rejection should be withdrawn.

The application is believed to be in condition for allowance. Prompt notification of the same is earnestly sought.

Respectfully submitted,

Potomac Patent Group PLLC

Date: November 15, 2007

By: /John F. Guay, Reg. No. 47,248/
John F. Guay

P.O. Box 270
Fredericksburg, Virginia 22404
703-718-8884